

|  |
| --- |
| Estimate Performance and Capacity Requirements for Microsoft Project Server 2010  This document is provided "as-is". Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.  Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.  This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.  © 2010 Microsoft Corporation. All rights reserved. |

# 

Introduction

This performance and capacity planning document provides guidance on the footprint that usage of Microsoft Project Server 2010 has on topologies running Microsoft SharePoint Server 2010.

For general information about how to plan and run your capacity planning for SharePoint Server 2010, see the TechNet article: “[Plan for performance and capacity (Office SharePoint Server)](http://technet.microsoft.com/en-us/library/cc262971.aspx)”.

Intended Audience

This document is intended as a starting point for helping individuals who are planning to deploy Project Server 2010 to identify their requirements, and to design an appropriate hardware and software topology to match those requirements.

The document assumes you have knowledge about the basic structure and functionality of a Project Server 2010 deployment. The [Project Server 2010 SDK](http://go.microsoft.com/fwlink/?LinkId=191049) is an excellent starting point for learning more about the basic architecture of Project Server 2010.

Contents

[Capacity Planning Strategy 5](#_Toc260815244)

[Estimating throughput targets 5](#_Toc260815245)

[Typical Datasets 7](#_Toc260815246)

[Other Variables to Consider 8](#_Toc260815247)

[Recommendations 9](#_Toc260815248)

[Hardware Requirements and Recommendations 9](#_Toc260815249)

[Small Dataset Hardware Recommendations: 9](#_Toc260815250)

[Medium Dataset Hardware Recommendations: 12](#_Toc260815251)

[Large Dataset Hardware Recommendations: 16](#_Toc260815252)

[Virtualization Recommendations 18](#_Toc260815253)

[Network Requirement 19](#_Toc260815254)

[Browser Requirement 19](#_Toc260815255)

[Scaled-up and Scaled-out Topologies 19](#_Toc260815256)

[To provide for more user load: 20](#_Toc260815257)

[To provide for more data load: 20](#_Toc260815258)

[Recommended server role ratio: 21](#_Toc260815259)

[Isolating from SharePoint Server: 21](#_Toc260815260)

[Investing rules of thumb: 21](#_Toc260815261)

[Optimizations 22](#_Toc260815262)

[Baselining 22](#_Toc260815263)

[Database Server Optimizations 22](#_Toc260815264)

[Master Projects 22](#_Toc260815265)

[Security Setting Optimizations 22](#_Toc260815266)

[Views Optimizations 23](#_Toc260815267)

[Custom Field Optimizations 23](#_Toc260815268)

[Local Custom Fields 24](#_Toc260815269)

[Page Payload Optimizations 24](#_Toc260815270)

[Timesheet Setting Optimizations 24](#_Toc260815271)

[Queue Optimizations 24](#_Toc260815272)

[Workload and Process Optimizations 25](#_Toc260815273)

[Workflow Optimizations 25](#_Toc260815274)

[Custom Solution (Programmability) Optimizations 26](#_Toc260815275)

[Performance monitoring 26](#_Toc260815276)

[Web servers 26](#_Toc260815277)

[Database servers 27](#_Toc260815278)

[Queue Counters 27](#_Toc260815279)

[Common bottlenecks and their causes 29](#_Toc260815280)

[See Also 31](#_Toc260815281)

# Capacity Planning Strategy

## Estimating throughput targets

Many factors can affect throughput. These factors include the number of users; the type, complexity, and frequency of user operations; the number of postbacks in an operation; and the performance of data connections. Each of these factors can have a major impact on farm throughput. You should carefully consider the factors discussed in this section when you plan your deployment.

Microsoft Project Server 2010 can be deployed and configured in a wide variety of ways. As a result, there is no simple way to estimate how many users can be supported by a given number of servers. Therefore, make sure that you conduct testing in your own environment before you deploy Project Server 2010 in a production environment.

When undertaking your capacity planning for Microsoft Project Server 2010, you need to be aware of the variables that can affect the performance of a Project Server deployment.

Because of the rich functionality set that Project Server provides, deployments that seem similar when described at a high level can differ substantially in their actual performance characteristics. It’s not enough to characterize your demands by just the number of projects, or the number of users you will have in the system. Thinking about the performance of your Project Server deployment requires a more nuanced and holistic approach. For example, workloads, and subsequently your hardware needs, will differ in relation to the following variables:

|  |  |
| --- | --- |
| Projects | * *Number of projects* * *Typical project sizes in terms of tasks* * *Number of project level custom fields* * *Level of linking (dependencies) between tasks* |
| Users | * *Concurrency of users*. How many users will be hitting the system at the same time? What is the average load, what are the spikes in traffic? * *What security permissions do users have*? This affects both the amount of data the server needs to present to the user at a given time, along with the complexity of the security checks the server has to do. * *Geographic distribution of users*. When users are spread over large geographical areas there can be detrimental performance effects due to network latency. This also impacts usage patterns insofar as users are likely to hit servers at different times during the day, making it harder to find low-traffic periods in which to run maintenance tasks such as backups, reporting, or Active Directory sync. |
| Usage Patterns | * *Workload conditions*. Which set of features are being commonly utilized? For example, a deployment that uses time-sheeting heavily will have different characteristics than one that does not use time-sheeting. * *Average time between page requests*. * *Average session time.* * *Payload of Pages* (How many web parts do you have on a given page? How much data do they contain?) |

To help you in your capacity planning this document describes three data sets, which we have found to characterize small, medium and large Project Server deployments. For each of these data sets, we then recommend one of three “rule-of-thumb” hardware topologies that should approximately satisfy the needs of similar datasets. With these starting point topologies in mind, we highlight factors that may require you to adjust these hardware topologies, outlining how you should evaluate if you need to decrease or increase the allocated resources to scale to your particular needs.

The approach you should take in your capacity planning is as follows:

1. Determine which of the datasets (small, medium, or large), is the closest match to your expected dataset. This is covered in the section of this document.
2. Use the hardware topology we recommend for that dataset size as a starting point approximation for what your needs will be. The recommended topologies are covered in the section.
   * Once again, it is important to realize that the needs of your particular dataset and usage patterns may require either fewer or more hardware resources than that approximate topology. The Recommendations section goes into depth about how you can assess whether or not you should add additional resources to the topology, and where to add them.
3. Monitor your application’s performance using the guidelines we outline in the section. In that section, we specify the key metrics you will want to track to determine when and how you need to scale your topology.
4. Optimize your deployment according to the suggestions provided in the section of this document.
5. Depending on your chosen topology, your dataset, your usage patterns, and the performance metrics you observe, follow the recommendations on scaling given in the following sections:

* :
  + This section gives advice regarding the type of strategy you should pursue when scaling depending on your current needs. Should you purchase additional servers, or should you purchase additional resource capacity (memory, CPU, disk) for the servers you already have?
* :
  + This section describes the likely source of bottlenecks in your system, how you might spot them through monitoring, and how issues related to these bottlenecks can commonly be resolved.

## Typical Datasets

The datasets described in this section are characterized by the variables listed and explained in the table below. These variables may not capture all of the factors that affect Project Server’s performance (e.g. they do not capture the mix of features you tend to use in your deployment). However, they do capture much of the information that is significant in determining appropriate capacity.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Entity** | **Description/Notes** | **Small** | **Medium** | **Large** |
| 1 | Projects |  | 100 | 5000 | 20000 |
| 1 | Tasks |  | 17125 | 856250 | 3425000 |
| 1 | Avg. Tasks Per Project |  | 171.25 | 171.25 | 171.25 |
| 2 | Task Transaction History | The number of times status tends to be submitted and approved for any given task | 10 | 100 | 1000 |
| 1 | Assignments |  | 22263 | 1113125 | 4500000 |
| 1 | Avg. Assignments Per Task |  | 1.3 | 1.3 | 1.3 |
| 2/3 | Approvals | Pending updates per manager | 50 | 600 | 3000 |
|  | Users |  | 1000 | 10000 | 50000 |
| Custom Fields | Project (Formula) |  | 3 | 20 | 25 |
| Project (Manual) |  | 2 | 40 | 50 |
| Task (Formula) | Task formula fields tend to take the largest toll on performance because they need to be computed for each task. | 6 | 12 | 15 |
| Task (Manual) |  | 4 | 8 | 10 |
| Assignment Rolldown |  | 50% | 50% | 50% |
| Resource |  | 10 | 20 | 25 |
| Look up Table Custom Fields |  | 2 | 15 | 100 |
| 1 | Timesheets (per year) | The more you utilize Timesheets, the more resource demands will be placed on the SQL Server. | 52000 | 780000 | 8,320,000 |
| 1 | Timesheet Lines |  | 5 | 10 | 10 |

*Note:* in our dataset size descriptions, the number of custom fields includes only the enterprise custom fields, not department custom fields. Department custom fields have essentially the same impact on the performance of Project Server 2010 as enterprise custom fields do. Therefore, if you have a large number of departmental custom fields (especially at the task level) you will require additional resources to support this. The prescriptions that are made regarding custom fields throughout this document apply to both enterprise custom fields and department custom fields.

### Other Variables to Consider

**Concurrency of Users:**

Concurrent user load is often a significant factor in setting capacity requirements. You may have fewer users in the system, but they may all transact with the server simultaneously during your “peak” traffic periods. For example, an organization that has its users all submit status/timesheet updates at the same time of the week will likely notice a substantial decrease in performance during those periods. If you have heavy peak usage periods, you will need to add additional resources to the topology recommended for your dataset.

**Split of User Roles:**

The distribution of your users between Administrators, Portfolio Administrators, Project Managers, and Team Members, will affect the performance of your deployment insofar as each type of user has access to a varying amount of data. Users in different security categories may vary with regard to how many projects and resources they may be able to see. Administrators, for example, will be able to see all projects on the server when loading Project Center, and all resources when they load Resource Center. In comparison, a Project Manager may be able to see only his own projects. The result is that these users may be subject to a diminished perceived performance. Where possible, we suggest you limit the number of projects, tasks or resources shown in a given view by defining appropriate filters in the views you define in Server Settings>Manage Views.

**Global Distribution of Users:**

**Issues, Risks and Deliverables**

Having larger numbers of these entities may place additional load on your SQL Server. In particular, it is the act of viewing and interacting with these entities in the Project Site that is likely to create the additional load. If you use these features heavily, you may want to allocate additional resources to your SQL Server to maintain a high-level of performance. Given that these artifacts and the Project Site functionality are SharePoint sites and lists, for scaling these aspects of Project Server 2010 consult the documentation around scaling SharePoint Sites and Lists:

**Calendars:**

Custom calendars can be defined for projects, tasks and resources. These largely impact the scheduling engine, exerting higher CPU on the Application and Database Servers.

# Recommendations

This section provides general performance and capacity recommendations. Use these recommendations to identify a suitable starting topology for your requirements, and to decide whether you have to scale out or scale up the starting topology.

Throughout this section we refer to the three different roles: the Web Front End Server Role, Application Server Role, and Database (SQL) Server Role. These are all components of a complete Project Server 2010 deployment. The Web Front End Servers act as the interface for users accessing Project Server. The Application Server handles the requests to the data tier of Project Server, and implements the business logic of Project Server 2010. Lastly, the database tier is the data source, housing the Project Server 2010 databases. For small deployments the Web Front End Server, Application Server and Database Server roles may be combined on the same physical machine. For larger deployments it may be necessary to separate these onto separate machines, even having multiple physical machines acting in the same role. We discuss this in more detail in the section of this document. To learn even more about Project Server 2010’s architecture please refer to the [Project Server 2010 Architecture](http://msdn.microsoft.com/en-us/library/ee767687(v=office.14).aspx#pj14_Architecture_PWAServer) section of the Project Server 2010 SDK.

## Hardware Requirements and Recommendations

Project Server 2010 requires an installation of Microsoft SharePoint Server 2010. The SharePoint Server 2010 hardware and software requirements are available at:

<http://technet.microsoft.com/en-us/library/cc262485(office.14).aspx>

This section suggests a minimum requirement and a recommended topology for each of the small, medium and large dataset sizes characterized in the section. The minimum requirement is a topology that will be functional with the given data set, but may suffer from substantially decreased perceived performance. The recommended topologies for each dataset should be sufficient for obtaining reasonable performance with most usage patterns on those dataset sizes. However, we encourage you to take into account the specific recommendations given throughout the rest of this document for determining if you will need to expand beyond the topology that is recommended for your approximate dataset. In general, you should monitor the performance metrics of your topology, and scale it accordingly if you are unsatisfied with the performance characteristics.

### Small Dataset Hardware Recommendations:

#### Minimum Requirement:



A single server that acts in all three roles: Application Server, Web Front End, and Database (SQL) Server. This server contains both the SharePoint Content Database, along with the four Project Server Databases.

|  |  |
| --- | --- |
| **Component** | **Minimum requirement** |
| Processor | 64-bit, four-core, 2.5 GHz minimum per core |
| RAM | 4 GB for developer or evaluation use  8 GB for single server and multiple server farm installation for production use |
| Hard disk | 80 GB for installation  For production use, you need additional free disk space for day-to-day operations. Add twice as much free space as you have RAM for production environments. |

#### Recommended:

Please note that because Project Server 2010 coexists with SharePoint Server 2010, it will generate additional resource usage (Processor, RAM, and Hard Disk). The guideline requirements for SharePoint Server 2010 are also valid for a Project Server 2010 installation *with a small data set and light usage*. However, for more substantial data sets and usage patterns, additional hardware resources will be required. For a single-box deployment, with a small data set, 16GB of RAM is advised to assure a high level of perceived performance.

Beyond this, if possible, we recommend that you separate your SQL Server from the Application and Web Front End tiers by placing your databases onto a dedicated SQL Server. The diagram below illustrates this type of topology. For instructions on how to do this please refer to: [Deployment for Project Server 2010](http://technet.microsoft.com/en-us/library/cc197280(office.14).aspx)



|  |  |
| --- | --- |
| **Web Front End / Application Server** | |
| **Component** | **Recommended** |
| Processor | 64-bit, four-core, 2.5 GHz minimum per core |
| RAM | 4 GB for developer or evaluation use  8 GB for single server and multiple server farm installation for production use |
| Hard disk | 80 GB |

|  |  |
| --- | --- |
| **SQL Server** | |
| **Component** | **Recommended** |
| Processor | 64-bit, four-core, 2.5 GHz minimum per core  (if your dataset size is considerably larger than the medium dataset, 8 cores are recommended) |
| RAM | 16 GB |
| Hard disk | 80 GB  For general prescriptions around planning your Storage and SQL Server, and optimizing them for performance, please consult the [Storage and SQL Server capacity planning and configuration (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/cc298801(office.14).aspx) |

### Medium Dataset Hardware Recommendations:

#### Minimum Requirement:

For a medium dataset, we suggest that as a minimum you have a dedicated Web Server, Application Server and SQL Server. This is what is referred to as a 1 X 1 X 1 topology, and it is illustrated in the diagram below. The recommended hardware specifications for each of the servers are stated following the diagram. The section of this document gives more instructions on how to achieved this separation of the tiers across multiple servers



|  |  |
| --- | --- |
| **Web Front End Server** | |
| **Component** | **Minimum requirement** |
| Processor | 64-bit, four-core, 2.5 GHz minimum per core |
| RAM | 4 GB for developer or evaluation use  8 GB for single server and multiple server farm installation for production use |
| Hard disk | 80 GB |

|  |  |
| --- | --- |
| **Application Server** | |
| **Component** | **Minimum requirement** |
| Processor | 64-bit, four-core, 2.5 GHz minimum per core |
| RAM | 4 GB for developer or evaluation use  8 GB for single server and multiple server farm installation for production use |
| Hard disk | 80 GB |

|  |  |
| --- | --- |
| **SQL Server** | |
| **Component** | **Minimum requirement** |
| Processor | 64-bit, four-core, 2.5 GHz minimum per core  (if your dataset size is considerably larger than the medium dataset, 8 cores are recommended) |
| RAM | 16 GB |
| Hard disk | 100 GB  For general prescriptions around planning your Storage and SQL Server, and optimizing them for performance, please consult the [Storage and SQL Server capacity planning and configuration (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/cc298801(office.14).aspx)  At this scale, for the SQL Server some important considerations also mentioned in that document are:   * Ideally, you should separate and prioritize data among disks. Place your data files and your SQL Server 2008 transaction logs on separate physical hard disks * RAID 5 should provide a good compromise between reliability, and throughput. |

With a small dataset deployment, it is likely acceptable to have both your SharePoint Server 2010 content databases, and your Project Server 2010 databases (draft, published, archive, and reporting), on the same dedicated SQL Server.



In some cases people use the SharePoint Server 2010 instance that Project Server is installed on top of as their primary SharePoint Server deployment. For small databases this may be a feasible approach, but is not recommended for medium and large databases. If the SharePoint Server 2010 instance which Project Server 2010 is coexisting with also gets heavy usage (i.e. you are not using that SharePoint Server 2010 deployment specifically for Project Server 2010 functionality), then separate the four Project Server 2010 databases from the SharePoint Server 2010 content databases, placing them on their own dedicated SQL Server. For more information on how to do this please refer to: [Deployment for Project Server 2010](http://technet.microsoft.com/en-us/library/cc197280(office.14).aspx).

#### Recommended:

The minimum requirements specified for medium sets can be scaled-out and scaled-up to handle additional load. The scaled-up and scaled-out topologies discuss considerations about how to handle increased user load, and increased data load. If your dataset is substantially larger than the medium dataset or you heavily use some of the features that generate additional data load, such as time-sheeting, then it is advisable for you to take some of the measures described in the section to handle your additional user and data load.

As a general prescription, you will want to be able to prepare for the needs to handle additional user load and data load by having sufficient machines to add additional Web Front End servers and Application Servers to your topology. The hardware specifications of your Web Front End servers and Application servers can remain largely the same. A 4 x 2 x 1 topology should be sufficient for handling the needs of most medium data sets and usage patterns (depicted below).



However, scaling out your Application and Web Front End servers will add additional load on your SQL Server, which you will need to compensate for by adding additional memory and CPU resources. The following SQL Server specification should be able to handle the performance needs of most medium datasets. You should scale to handle additional data load as recommended in the section.

|  |  |
| --- | --- |
| **SQL Server** | |
| **Component** | **Minimum requirement** |
| Processor | 64-bit, **eight-core**, 2.5 GHz minimum per core  (if your dataset size is considerably larger than the medium dataset, 8 cores are recommended) |
| RAM | **32 GB** |
| Hard disk | 160GB  For general prescriptions around planning your Storage and SQL Server, and optimizing them for performance, please consult the [Storage and SQL Server capacity planning and configuration (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/cc298801(office.14).aspx)  At this scale, for the SQL Server some important considerations also mentioned in that document are:   * Ideally, you should separate and prioritize data among disks. Place your data files and your SQL Server 2008 transaction logs on separate physical hard disks * RAID 5 should provide a good compromise between reliability, and throughput. |

In general, the best way to identify if the topology you have designed satisfies your performance needs is to set up a staging environment to test your topology, and monitor the performance characteristics. Please refer to both the [Performance Monitoring](#_Performance_monitoring_1) section of this document, and the [Microsoft Project Server 2010 Performance Testing Lab](http://technet.microsoft.com/en-us/library/ee956502(office.14).aspx) .

### Large Dataset Hardware Recommendations:

For large datasets, you will find that the data load will be the most substantial performance bottleneck.

Generally, at a minimum for large datasets you will want a 4 x 2 x 1 topology. The hardware characteristics of the Web Front End and Application Servers can generally remain the same as those recommended for the small and medium datasets. However, given that your SQL Server will be your bottleneck, you may find that this constrains your ability to scale out to additional Web Front End and Application Servers. If you find that data load is your bottleneck, you may find that additional Web Front End and Application severs do not produce an improvement in throughput.

For large datasets, if the SharePoint Server 2010 instance which Project Server 2010 is coexisting with is also getting heavy usage (i.e. you are not using that SharePoint Server 2010 deployment specifically for Project Server 2010 functionality), then it is recommend to separate the 4 Project Server 2010 databases from the SharePoint Server 2010 content databases, placing them on their own dedicated SQL.

Given that your SQL Server will be your bottleneck, you will want to invest in additional resources on the SQL Server tier of your topology. You can “scale-up” your SQL Server by adding additional RAM, CPU and Hard Disk resources.

Below we list the minimum and recommended specifications for the SQL Tier of a large dataset topology.

#### Minimum Requirement:

|  |  |
| --- | --- |
| **SQL Server** | |
| **Component** | **Minimum requirement** |
| Processor | 64-bit, **eight-core**, 2.5 GHz minimum per core  (if your dataset is considerably larger than the medium dataset, 8 cores are recommended) |
| RAM | **32 GB** |
| Hard disk | 250 GB  For general prescriptions around planning your Storage and SQL Server, and optimizing them for performance, please consult the [Storage and SQL Server capacity planning and configuration (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/cc298801(office.14).aspx)  At this scale, for the SQL Server some important considerations also mentioned in that document are:   * Ideally, you should separate and prioritize data among disks. Place your data files and your SQL Server 2008 transaction logs on separate physical hard disks * RAID 5 should provide a good compromise between reliability, and throughput. |

#### Recommended:

|  |  |
| --- | --- |
| **SQL Server** | |
| **Component** | **Minimum requirement** |
| Processor | 64-bit, **eight-core**, 2.5 GHz minimum per core  (if your considerably larger than the medium dataset, 8 cores are recommended) |
| RAM | **64 GB** |
| Hard disk | 300 GB or more  *Separate your reporting database onto a separate database server.*  At this scale, for the SQL Server some important considerations also mentioned in that document are:   * Ideally, you should separate and prioritize data among disks. Place your data files and your SQL Server 2008 transaction logs on separate physical hard disks * RAID 5 should provide a good compromise between reliability, and throughput. |

### Virtualization Recommendations

Project Server 2010 does support running on virtualized machines. Most of the advice given for virtualization of SharePoint Server 2010 also applies to Project Server 2010. For documentation on virtualization in SharePoint Server 2010 refer to [Virtualization planning (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/ff607968(office.14).aspx) in the TechNet Library. You may also refer to the Project Server 2007 Virtualization guide for additional information on virtualization and Project Server 2010, as most of that guidance is still applicable. However, as in any situation where virtualization is employed, it is important to consider contention for physical machine’s resources between virtualized machines running on the same physical instance.

**NOTE**: We do not recommend running SQL Server on a virtualized machine. The competition for resources on a virtualized machine can substantially decrease the performance of the SQL Server. If you must run your SQL Server in a virtualized environment, we recommend that you use the following settings:

* **Network Adapter:** if using Hyper-V virtualization, you should utilize the virtual network adapter rather than the legacy network adapter.
* **Virtual Disk:** 
  + For the virtual machine that you are running SQL Server on, we recommend that you select the “pass through” option for the disk type (rather than dynamic, or fixed). If this is not an option, you should utilize a fixed disk size rather than a dynamically sized virtual disk.
  + We recommend that you select IDE over SCSI for your boot drive.
  + Allocate sufficient hard disk space to handle the expected maximum size of your dataset and ULS logging demands.
* **Memory**:
  + You should allocate as much memory to the virtual machine running SQL Server as can feasibly be allocated. This should be comparable to the amount of memory required/recommended for physical boxes serving the same function.
  + You need to also account that some memory needs to be reserved for the Host Operating System. At least 2GB of memory should be reserved for the Host Operating System.

Running the Web Front End or Application Servers in virtualized environments tends to not be as detrimental to the performance of running SQL Server in a virtualized environment.

### Network Requirement

For most Project Server deployments, network bandwidth tends to not be the bottleneck on performance. The table below lists the recommended specifications of network components

|  |  |  |
| --- | --- | --- |
| **Component** | **Small and Medium** | **Large** |
| # of NICs | 1 | 2 |
| # NIC Speed (Network) | Any speed greater than 100mbps should be fine | 1 Gb/s |
| Load Balancer Type | NLB or hardware, both are acceptable | NLB or hardware, both are acceptable |

A general aim should be to try to maintain low latency between the Application and SQL Server tiers.

### Browser Requirement

Please note that currently we only support Internet Explorer 7 and Internet Explorer 8 browsers.

## Scaled-up and Scaled-out Topologies

To increase the capacity and performance of your topology you can do two things. You can scale up by increasing the capacity of your existing server computers and scale out by adding additional servers to the topology.

Monitoring resource usage on your Project Server 2010 deployment servers can give insights into which strategy you will want to pursue: scaling-up vs. scaling-out. The performance metrics that can help inform your decisions are discussed in the [Performance monitoring](#_Performance_monitoring) section of this document.

The physical servers that you deploy Project Server 2010 on will play the following server roles:

* + Web Front End
  + Application Server
  + Database (SQL) Server

In a single-machine deployment, one physical server plays all three of these roles. You can scale out by dividing ownership of these roles amongst different physical machines (optionally, they could also be virtual machines; please read the section for additional guidelines around virtualization). The first step you should take when starting to scale out is to separate the Database (SQL) Server role onto its physical machine, while having the other physical machine act as the Web Front End and Application Server.

**Note**:

Project Server 2010 only provides limited scale-out capabilities. While additional servers can be added to act as Web Front End servers, and Application servers, on the backend the SQL Server has limited scale-out possibilities. At most, you can separate Project Server databases onto two dedicated SQL Servers – one holding the reporting database, and the other housing the archive, draft, and published databases. What this implies is that for large datasets the SQL Server becomes the primary bottleneck, and a primarily “scale-up” strategy needs to be employed, where additional hardware resources are added to the SQL Server.

### 

### To provide for more user load:

* Scale-out by adding more Web Servers to serve as dedicated Application and Web Front End servers.
* Note that as you add more Web Front End servers, and Application servers, the load on the SQL Server will increase, and your SQL Server could in turn become your bottleneck.
* You may also scale up any Web Front End or Application Server to improve performance by increasing the capacity of those servers.

### To provide for more data load:

* To provide for more data load, add capacity to the database server by increasing the capacity of that single server.
* Separate the Project Databases from the SharePoint Databases by moving the four Project databases onto their own dedicated database server.
* Beyond this, it is also possible to detach the Reporting Database to a separate database server. The server with the Reporting database can then additionally act as a failover for the server with the other three Project databases. It is possible to achieve this by using SQL mirroring, and setting up a cross-fail relationship between the server housing the Reporting Database, and the server housing the remaining Project databases.
* Note: Project Server 2010 does not support scaling out the Database component through SQL replication. While it is possible to perform SQL mirroring on a Project SQL Server for the purposes of backing up data, Project Server 2010 is unable to take advantage of SQL replication to reduce read loads on the SQL Server.

### Recommended server role ratio:

As a rule of thumb, a recommended ratio for maintaining a manageable load on the SQL Server is:

2 Web Front Ends : 1 Application Server : 1 SQL Server

*Note:*

* This recommended ratio might not apply for deployments utilizing more robust hardware, especially for the database server.
* The recommended ratio also varies based on the data set size, and the usage patterns. For example, larger data sets would constrain the ability to fan-out, requiring a smaller ratio of Web Front Ends and Application Server to the SQL Server. In terms of usage patterns, one example is that a deployment of Project Server 2010 that heavily utilizes Time-sheeting functionality, will also constrain this ratio, as Time-sheeting will consume substantial resources on the SQL Server.
* For some cases with particularly high data loads, you may need to even have 1:2:2 ratio to maintain high performance characteristics.

### Isolating from SharePoint Server:

* As already mentioned, it is possible to separate the Project Server databases from the SharePoint databases, by placing them on their own dedicated database server.
* It is also possible to separate at the Application Tier level. While Project Server is a SharePoint service, it is possible to have that application instance run on a dedicated server.

### Investing rules of thumb:

* As a rule of thumb, in the early stages of scaling your Project Server deployment, you will want to invest primarily in purchasing additional memory. Most often, the subsequent areas you would want to invest in are disk spindles, and then network resources.

## Optimizations

### Baselining

Generally, it is recommended that you limit the number of baselines that you have saved at any given time. There is a hard limit of only 11 baselines supported at a given time.

### Database Server Optimizations

Given that Project Server 2010 is a data intensive application, optimizing your database tier can add considerable gains to performance. Please refer to [SQL Server and Storage Capacity Planning and Configuration](http://officecpub/Teams/itpro/Documents/Technical%20Reviews/Enterprise%20Server/O14_Manage%20Databases/Storage%20and%20SQL%20Server%20capacity%20planning%20and%20configuration.docx) for broad guidelines on optimizing your SQL Server settings. Some of the recommendations below highlight those made in the SQL Server documentation:

* Separate the database files and the transaction log files away from the OS drives – preferably each to their own partition. This helps by reducing IO contention between the host operating system and SQL Server, and then also between SQL database files and log files, which tend to have different update patterns depending on what recovery strategy is used.
* SQL Server supports the use of statistics. Ensuring that the statistics are up-to-date will improve the ability of the SQL Query Optimizer.
* Ensure that your SQL Server indexes are not fragmented.
* Separate the TempDB onto its own partition. Split the database into several physical files – ideally, splitting it into as many files as you have processors on your database server.
* Consider utilizing a RAID subsystem for your data needs
  + RAID 5 is recommended for medium data set sizes.
  + RAID 5 is acceptable for large dataset sizes, but RAID 10 is ideal
* Move indexes onto their own partition
* Project Server has been optimized to utilize the benefits of SQL CLR. Although SQL CLR is not a requirement, if available and enabled, it has the potential for increasing the performance of some operations

### Master Projects

When using the Master Projects functionality in Project Server, note that changes in the Master Project schedule will have an impact on the schedules of subprojects within the Master Projects. Therefore, schedule changes for very large Master Projects may execute slowly, because their subproject plans may need to be updated.

### Security Setting Optimizations

The security settings you use for your users can have a significant impact on performance characteristics because they determine both the amount of data users load when viewing projects, and also the complexity of the security checks that are performed to determine which sets of data users have permissions on.

For example, Administrators will have access to all of the Projects you have stored in Project Server, and are therefore required to load all of the data when interacting with them. Team members may not need access to all the data, so we can limit the amount of data that is sent to them via security categories:

* Use groups and categories where possible rather than more granular permissions that require additional complexity in the security checks
* Try to restrain people’s security permissions to the projects they need to have access to, that way they load only the data they need to when interacting with Project Server

### Views Optimizations

As in the case of security settings, you should attempt to limit the data that is presented to users by restricting the number of columns in a given view to only the columns that users with permission on that view need to see. Also, note that adding Custom Field columns will have a more detrimental impact on view performance.

* You may also use filters to limit the amount of data that must be loaded when loading a particular view. Be aware, however, that filters with complex logic will require additional computation, and may as a result slow down performance.

### Custom Field Optimizations

The performance impact of custom field usage depends on several aspects of the custom fields that are used (both Department and Enterprise custom fields). The following are some considerations and suggestions regarding the performance aspects of custom fields:

* The performance impact of your custom fields will depend on:
  + The quantity of data stored in the custom fields you use (are they generally sparse, or are there large quantities of data in a given custom field column?)
  + For formula fields, the more complex the formulas employed, the more substantial the negative impact on performance will be.
  + The level the custom fields occur at:
    - There are usually far more tasks than projects in a dataset, and therefore, custom fields applied at the task level will have a more substantial negative impact on performance than project level custom fields.
* Generally, the prescription is to try to limit the number of custom fields used, especially at the task level.
  + As a rule of thumb, try to use less than 10 -15 task level Enterprise Custom Fields.
* Task and assignment custom fields are the primary bottleneck in saving from WinProj to the server in most observed customer data sets.

### Local Custom Fields

In line with the recommendations around optimizing custom field use, local formula field use can also be optimized by limiting the number of local formula fields used in the Project client.

* In particular, limit the use of formula fields where possible, as they require additional data transfer that will increase the time required for saving to the server.

### Page Payload Optimizations

One of the most important factors in determining a given page’s load time is the amount of data that needs to be accessed on a given page request. This is largely determined by the number of Web Parts, and the types of Web Parts and how much data they present on a given page.

The following are some general recommendations on limiting the payloads of your Project Server pages:

* Limit the number of web parts on a given page
  + Limit the amount of data that these Web Parts load to only the data that they need to be loading.
  + Payload considerations are particularly important for Project Detail Pages (PDPs), where there tend to be a larger number of Web Parts on a given page, and more customization occurs.
* If you do not benefit from the Notifications Web Part, an Administrator should remove it from the shared page layout.

### Queue Optimizations

Project Server 2010 utilizes a queuing system to handle how it services requests, enabling it to serve a greater number of requests overall. Certain settings around how the queue operates can be altered through the Server Settings > Queue > Queue Settings page. This section gives a brief explanation of the settings you can modify, and how to optimize them for your needs.

* Max Number of Threads (1-20, default 4):  This determines how many jobs the queue can process in parallel at any given time.  Note that this applies to all machines in the farm—if you have 3 Application servers, and you set this value to 4 for the project queue, you can process up to 12 independent project jobs at a time.
* Polling interval (500-300,000, default 1000):  The interval at which the queue checks for new jobs, measured in milliseconds.
* Fast Polling (default ON):  This determines whether the queue waits for the next time interval before processing another job.  When Fast Polling is OFF, jobs are only selected for processing once per interval, so if there is a sequence of jobs that take 0.1 second each, and your polling interval is 1 second, maximum throughput will be roughly 1 job/second per thread.  When fast polling is ON, the queue will check for a new job immediately after completion.  Thus, the same sequence of jobs would achieve a throughput of roughly 10 jobs/second per thread.

If you find that queued jobs are taking an excessive amount of resources away from a synchronous workload, you can try the following:

1. If you have a large number of jobs that are processed in parallel (that is, you see multiple jobs in the “processing” state at the same time when you check the queue status), you can try reducing the thread count.
2. If (1) has no effect, or your workload is *not* highly parallel (only one job in the “processing” state at a time), and most of your jobs are relatively short-running, you can turn off Fast Polling.
3. Finally, you can increase the polling interval if (1) and (2) prove insufficient.

For example, if you determine that multiple users are saving projects from the Project Professional client at once, and that this is thrashing your system, you could reduce parallelism on the Project Server queue.  If you have several smaller jobs that happen in the same correlation, for example a large Windows SharePoint Server sync (e.g. due to starting an Active Directory sync), or roll-up calculations for Timesheet reporting, you could benefit from turning off Fast Polling.

### Workload and Process Optimizations

Certain aspects of how you operate and maintain your Project Sever deployment can help improve the perceived performance of Project Server. This section covers a list of Business or IT related process modifications that can help to improve the perceived performance of your Project Server during periods when your users are most likely to interact with the system.

|  |  |
| --- | --- |
| Timesheeting and Status Submissions | If possible, try to stagger the times when users submit status updates and timesheets. This will act to reduce the strain on the system during peak periods by distributing the load over larger time intervals. |
| Backups | If possible, you should try to run backup processes during non-peak periods, as these are resource intensive processes that will diminish perceived performance for users attempting to use the system while they are running |
| Active Directory Sync | As with backup processes, you should try to run Active Directory Sync processes during non-peak periods, as these are resource intensive processes that will diminish perceived performance for users attempting to use the system while they are running |
| Reporting | As with backup processes, you should try to run the building of OLAP cubes for reporting during non-peak periods, as these are resource intensive processes that will diminish perceived performance for users attempting to use the system while they are running |

### Workflow Optimizations

When using the Workflows functionality, please be aware of the following actions that will take a toll on the performance of your deployment:

* It can take a long time to load the “Change or Restart Workflows” page in Server Settings when you have a large number of projects stored in the database.
* Restarting or changing the EPT for a large number of projects from Change or Restart Workflows page in Server Settings
* Having an approval process with a very large number of users.
* Having projects submitted at the same time from a workflow stage without check-in required

Generally, minimizing these actions, or executing them at low-traffic periods is recommended to optimize perceived performance.

### Custom Solution (Programmability) Optimizations

When developing custom solutions that interact with the Project Server 2010 PSI, please take into account the following performance recommendations:

* If you are deploying event handlers, be aware that the event handlers are synchronous. You should be careful when employing event handlers in your custom solutions, as if utilized ineffectively they can substantially increase the performance of Project Server.
* Your custom solution should try to minimize the calls it makes to queuing operations in the Project Server PSI. This is important, as you want to avoid loading the queue too deeply.
* For Line of Business (LOB) Applications, when automating data movement between Project Server and other applications, if you notice syncs with these types of applications are substantially degrading performance it is advised to run them during non-peak usage periods.
  + It is highly recommended that customers test and monitor their LOB application performance, in addition to their user-facing performance.
  + Where possible try to utilize intrinsic Project Server fields rather than custom fields to achieve the sync you desire between Project Server and the LOB applications.
  + Try to minimize the data you are moving between LOB applications and Project Server to the smallest subset you need for achieving the desired functionality.
* The [Project Server 2010 SDK](http://msdn.microsoft.com/en-us/library/ms512767(office.14).aspx), and related articles make further recommendations around maintaining high performance when developing custom solutions.

## Performance monitoring

To help you determine when you have to scale up or scale out your system, use performance counters to monitor the health of your system. Use the information in the following tables to determine which performance counters to monitor, and to which process the performance counters should be applied.

### Web servers

The following table shows performance counters and processes to monitor for Web servers in your farm.

|  |  |  |
| --- | --- | --- |
| Performance counter | Apply to object | Notes |
| Processor time | Total | Shows the percentage of elapsed time that this thread used the processor to execute instructions. |
| Memory utilization | Application pool | Shows the average utilization of system memory for the application pool. You must identify the correct application pool to monitor.  The basic guideline is to identify peak memory utilization for a given Web application, and assign that number plus 10MB to the associated application pool. |

### Database servers

The following table shows performance counters and processes to monitor for database servers in your farm.

|  |  |  |
| --- | --- | --- |
| Performance counter | Apply to object | Notes |
| Average disk queue length | Hard disk that contains SharedServices.mdf | Average values greater than 1.5 per spindle indicate that the write times for that hard disk are insufficient. |
| Processor time | SQL Server process | Average values greater than 80 percent indicate that processor capacity on the database server is insufficient. |
| Processor time | Total | Shows the percentage of elapsed time that this thread used the processor to execute instructions. |
| Memory utilization | Total | Shows the average utilization of system memory. |

### Queue Counters

The following table shows performance counters and processes to monitor for your Application Server.

|  |  |  |
| --- | --- | --- |
| Performance counter | Apply to object | Notes |
| % Sql Retries / Day | ProjectServer:QueueGeneral |  |
| Active Job Processing Threads | ProjectServer:QueueGeneral |  |
| Active Job Processing Threads | ProjectServer:QueueGeneral |  |
| Average Unprocessed Jobs / Day | ProjectServer:QueueGeneral |  |
| Current Unprocessed Jobs | ProjectServer:QueueGeneral |  |
| New Jobs / Minute | ProjectServer:QueueGeneral |  |
| Sql Calls / Hour/Day | ProjectServer:QueueGeneral |  |
| Sql Calls / Minute | ProjectServer:QueueGeneral |  |
| Sql Retries / Minute | ProjectServer:QueueGeneral |  |
| % Jobs Failed / Day | ProjectServer:QueueJobs |  |
| % Jobs Failed / Hour | ProjectServer:QueueJobs |  |
| % Jobs Retried / Day | ProjectServer:QueueJobs |  |
| % Jobs Retried / Hour | ProjectServer:QueueJobs |  |
| Average Processing Time / Day | ProjectServer:QueueJobs |  |
| Average Processing Time / Minute | ProjectServer:QueueJobs |  |
| Average Wait Time / Day | ProjectServer:QueueJobs |  |
| Average Wait Time / Minute | ProjectServer:QueueJobs |  |
| Jobs Failed / Minute | ProjectServer:QueueJobs |  |
| Jobs Processed / Hour/Day | ProjectServer:QueueJobs |  |
| Jobs Processed / Minute | ProjectServer:QueueJobs |  |
| Jobs Retried / Minute | ProjectServer:QueueJobs |  |
| ProjectServer:Winproj | Average time taken for Project Open |  |
| ProjectServer:Winproj | Percentage of incremental save to full save |  |
| ProjectServer:Winproj | Winproj full open count in the last hour |  |
| ProjectServer:Winproj | Winproj full save count in the last hour |  |
| ProjectServer:Winproj | Winproj incremental open count in the last hour |  |
| ProjectServer:Winproj | Winproj incremental save count in the last hour |  |
| ProjectServer:User Activity | PSI Calls per Second |  |

## Common bottlenecks and their causes

During performance testing, several different common *bottlenecks* were revealed. A bottleneck is a condition in which the capacity of a particular constituent of a farm is reached. This causes a plateau or decrease in farm throughput.

By monitoring your performance using the guidelines specified in the Performance Monitoring section, you may be able to better identify which bottlenecks are impacting the perceived performance of your Project Server deployment. The following table lists some common bottlenecks and describes their causes and possible resolutions.

|  |  |  |
| --- | --- | --- |
| Bottleneck | Cause | Resolution |
| Database contention (locks) | Database locks prevent multiple users from making conflicting modifications to a set of data. When a set of data is locked by a user or process, no other user or process can modify that same set of data until the first user or process finishes modifying the data and relinquishes the lock. | To help reduce the incidence of database lock contention you can:   * Scale up the database server. * Tune the database server hard disk for read/write.   Methods exist to circumvent the database locking system in SQL Server 2005, such as the NOLOCK parameter. However, we do not recommend or support use of this method due to the possibility of data corruption. |
| Database server disk I/O | When the number of I/O requests to a hard disk exceeds the disk’s I/O capacity, the requests will be queued. As a result, the time to complete each request increases. | Distributing data files across multiple physical drives allows for parallel I/O. The blog [SharePoint Disk Allocation and Disk I/O](http://go.microsoft.com/fwlink/?LinkId=129557) (http://go.microsoft.com/fwlink/?LinkId=129557) contains much useful information about resolving disk I/O issues.  Limit the number of projects and fields shown in a given view, so that limit the amount of data requested from the Database server.  Try to limit the number of custom fields you utilize, especially at the task level. Formula fields at the task level are particularly costly in terms of database server disk I/O when performing save operations from WinProj. |
| Web server CPU utilization | When a Web server is overloaded with user requests, average CPU utilization will approach 100 percent. This prevents the Web server from responding to requests quickly and can cause timeouts and error messages on client computers. | This issue can be resolved in one of two ways. You can add additional Web servers to the farm to distribute user load, or you can scale up the Web server or servers by adding higher-speed processors. Please |
| Server Memory Utilization | When you have a substantial number of large queue jobs executing, the server memory utilization can spike.  More complex server side scheduling calculations, or evaluation of formula custom fields, can also consume substantial memory resources.  As a result, the time to complete each request increases. | Monitor at which tier memory usage is a bottleneck: i.e. is the memory scarcity happening on the Application Server, the Web Front End Server, or the Database Server.  To resolve the lack of memory there are two options:   * Purchase and install additional memory for that tier. * Purchase additional application servers to handle the load. |
| Active Directory Sync | Project Server users and resources can be synchronized with the users of the service across multiple domains and forests. This feature helps administrators with tedious tasks, such as manually adding large numbers of users, updating user metadata such as email addresses, and deactivating users who no longer require system access. Active Directory synchronization can be done manually or on an automated schedule. The process of synchronization is resource intensive | It is best to run Active Directory Sync during non-peak user usage periods. That way, Active Directory Sync will not degrade the perceived performance of users.  In addition, try to avoid heavily nested groups, as they increase the complexity of the sync that must be performed, and result in longer sync processes. |
| Application Server CPU | Application Server CPU can be hit hard when:   * scheduling complex projects * evaluating formulas on complex projects, * running portfolio analyses on a large number of projects with Time-phased Resource Planning analysis turned on. | Monitor the Application Server’s CPU usage, and if it seems that it is utilizing a high percentage of its CPU resources, then add an additional Application Server to your topology to distribute the load.  Note that adding an additional Application Server will add additional threads that could cause increased load on the Database Server. This could create a new bottleneck on the database server, which may be resolved by allowing fewer Job Processor Threads in the Queue Settings. |
| Database Server CPU | Usually, database server CPU spikes when try to load views that comprise of a large number of projects, and a large number of fields shown. This will reduce the perceived user response time when that view is applied. | Limit the number of projects and the number of fields shown in a given view. |

## See Also

Project Server 2010 on TechNet: <http://technet.microsoft.com/projectserver/>

SharePoint 2010 Capacity Planning on TechNet: <http://technet.microsoft.com/library/cc262971>